Transferring Procedural Knowledge across Commonsense Tasks

Yifan Jiang[a;*], **Filip Ilievski**[a] **and Kaixin Ma**[b]

[a]Information Sciences Institute, Viterbi School of Engineering, University of Southern California
[b]Language Technologies Institute, School of Computer Science, Carnegie Mellon University

Abstract. Stories about everyday situations are an essential part of human communication, motivating the need to develop AI agents that can reliably understand these stories. Despite the long list of supervised methods for story completion and procedural understanding, current AI fails to generalize its procedural reasoning to unseen stories. This paper is based on the hypothesis that the generalization can be improved by associating downstream prediction with fine-grained modeling and the abstraction of procedural knowledge in stories. To test this hypothesis, we design LEAP: a comprehensive framework that reasons over stories by jointly considering their (1) overall plausibility, (2) conflict sentence pairs, and (3) participant physical states. LEAP integrates state-of-the-art modeling architectures, training regimes, and augmentation strategies based on natural and synthetic stories. To address the lack of densely annotated training data on participants and their physical states, we devise a robust automatic labeler based on semantic parsing and few-shot prompting with large language models. Our experiments with in- and out-of-domain tasks reveal insights into the interplay of architectures, training regimes, and augmentation strategies. LEAP's labeler consistently improves performance on out-of-domain datasets, while our case studies show that the dense annotation supports explainability.

1 Introduction

Building AI agents that understand stories is central to many domains, ranging from cooking [28] to science [12]. This is because practically any situation can be associated with a story that requires an agent to judge and explain its plausibility [7]. *How does one decide whether a story is plausible?* Let us consider the two similar stories shown in Figure 1. Story A makes sense because taking out the notebook is often followed by writing, and a key affordance of notebooks is to enable writing. Meanwhile, story B is implausible, as having the notebook at a different location hinders the possibility of writing in it. Thus, to understand stories about everyday situations, a model needs to be able to track the states of the relevant participants, understand the implications of described events, detect anomalous and unexpected behaviors, and project alternative and counterfactual scenarios [31].

While story comprehension has been a popular goal over the past decade [30], state-of-the-art methods typically lack pragmatic inference and focus solely on end-task goal prediction, which lacks the transparency of the intermediate reasoning process. The requirement of benchmark-specific training also limits their generalization to novel benchmarks and tasks. A parallel stream of research [28, 21] has developed methods for state tracking of participants in the domains of science and cooking to increase the model's interpretability. A recent task of procedural reasoning about physical processes [33] measures the ability of methods to simultaneously predict the plausible story, the conflicting sentence pairs for the implausible story, and the physical states of the participants. Although this task provides a natural bridge between the model's procedural understanding and the overall story assessment, it has only been considered in a supervised setting, raising questions about the generalizability of the findings on unseen data. Moreover, such fine-grained reasoning requires densely annotated stories, i.e., every participant's physical state need to be annotated for every step of the story. This naturally leads to high annotation costs and has limited the size of such datasets to a small scale. We hypothesize that training with fine-grained objectives (e.g., detecting conflicting sentences and modeling participants' physical states) can effectively prevent the model from learning shortcuts, therefore improving the generalization to unseen scenarios. To our knowledge, no existing effort has studied the transfer ability of such fine-grained procedural knowledge to unseen tasks, nor addressed the inherent lack of densely annotated data about story procedures.

In this paper, we *study the ability of AI models to transfer procedural knowledge across story-based tasks in a transparent manner*. Our contributions are as follows:

1. **A comprehensive framework called LEAP** (Learning from Experience by Annotating Procedures), that integrates state-of-the-art language model (LM) architectures, training regimes, and augmentation strategies. LEAP is designed to study the ability of models to transfer knowledge about procedures across story tasks.
2. **An automatic labeler** within LEAP that densely annotates collected stories based on semantic parsing and few-shot prompting. LEAP's labeler can be generically applied to annotate participants and their attributes in arbitrary realistic and synthetic stories.
3. **An extensive evaluation** of a wide range of models on representative procedural-driven tasks covering different transfer capabilities, task formats, and domains. We provide insights into the strengths and weaknesses of current models to reason over novel stories, and we showcase the ability of the developed labeler to generalize better with existing supervised methods.

We release our code and data[1] to support future research.

2 Related Work

Story Understanding has been a popular task over the past decade, resulting in many popular benchmarks [25, 18] and methods [19, 11].

* Email: {yifjia,ilievski}@isi.edu. kaixinm@andrew.cmu.edu.

[1] https://github.com/1171-jpg/LEAP

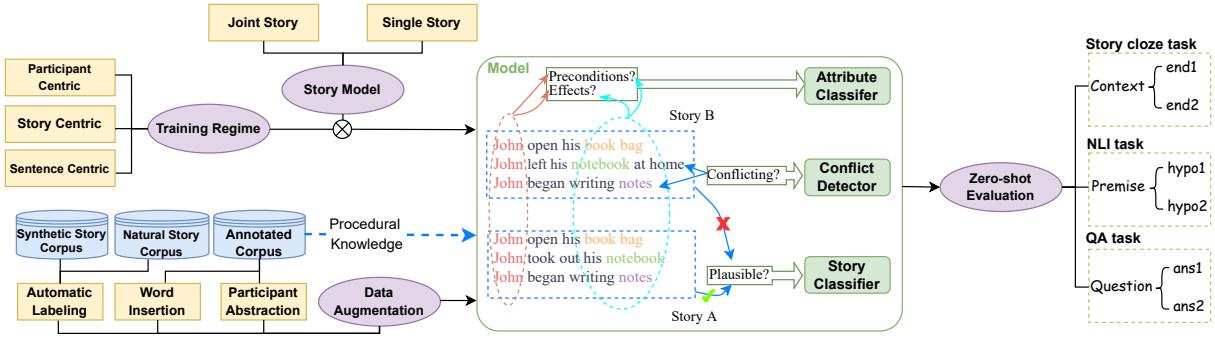


Figure 1: An illustration of the LEAP framework, which transfers procedural knowledge from a source task through zero-shot evaluation on unseen tasks. The model structure is presented in the green box. LEAP includes various story modeling strategies and training regimes. To address data sparsity, LEAP supports three data augmentation methods: participant abstraction, word insertion, and automatic labeling.

While some previous work has studied the effect of commonsense knowledge on story understanding [8], they only focus on the supervised setting and in-domain evaluation. Conversely, our paper focuses on the zero-shot evaluation setting and studies the transfer of procedural knowledge with different data augmentation strategies.

Procedural Reasoning Unlike story understanding task which only requires the final prediction label, procedural reasoning requires the model to track the states of the participants at every step in a process [5, 12]. Many dedicated methods have been developed for procedural reasoning [28, 38]. More recently, Storks *et al.* [33] proposed the TRIP benchmark to bridge the gap between procedural reasoning and story understanding, where the model needs to perform tiered reasoning over story pairs. Ma *et al.* [21] proposed a procedural reasoning model that can be extended to perform story understanding tasks. However, there has been no study of the zero-shot generalization ability of these aforementioned models.

In-Context Learning and Semantic Annotation With the recent progress of pre-trained large LMs (LLMs), prompting has become a popular approach to tackle many NLP tasks [6]. More specifically, zero-shot prompting directly feeds in the input to the model to elicit an output [35], while few-shot prompting additionally appends example demonstrations to the input to better guide the model's prediction [29]. Most of the previous work on prompting studied tasks that require simple outputs, e.g., classification [24], and only a few have attempted to apply prompting to semantic annotation tasks that require complex structured outputs. Spiliopoulou *et al.* [32] only focus on detecting the change of attributes and Madaan *et al.* [22] adapts LM annotation to in-domain data. In contrast, we not only explore prompting for procedural reasoning but also study the transfer ability of the elicited procedural knowledge to other tasks.

3 LEAP: Procedural Transfer Framework

We introduce a framework called **LEAP** (**L**earning from **E**xperience by **A**nnotating **P**rocedures) that enables the transfer of explicit procedural knowledge from a source task S to target tasks $T_1, T_2, ...$ in a zero-shot manner. S consists of pairs of plausible and implausible stories (P, P'). Each story comprises n sentences $s_1, s_2, ...s_n$ and is annotated with three procedural components: (1) the preconditions and the effects for each attribute $a \in A$ of each participant $e \in E$ at every step in the story (or E' for P', which is different from E); (2) a pair of conflicting sentences (s,s') in the implausible story; and (3) a plausibility label of the story, $P_{plau} \in \{0, 1\}$. A model that learned procedural knowledge rather than spurious correlations from the source task should be able to generalize to unseen tasks that require similar kinds of reasoning. To test this, we select target tasks T_i that are in multiple-choice format, where given a partial procedure description, a model has to select the answer that optimally completes the procedure.

The model structure, shown in green in the center of Figure 1, is loosely based on CGLI [21], which achieves state-of-the-art results on procedural understanding tasks. The architecture is based on a Transformer language model, which encodes a pair of stories in parallel and provides its encoding to three distinct output layers that are used to perform the stratified reasoning on three procedural components. An *Attribute classifier* predicts the preconditions and the effects for each attribute of each participant. A *Conflict detector* predicts the conflict sentences in the implausible story. A *Story classifier* determines which story is plausible. LEAP extends this general model with two story modeling methods, three training regimes, and three augmentation strategies. We also evaluate the LEAP variants on five out-of-domain tasks. A comprehensive overview of LEAP is shown in Figure 1.

3.1 Story Modeling

LEAP includes two ways of modeling stories: single and joint (as pairs). We will introduce the detail of the model structure and its special layers design for procedural components in *Single Story Model*. And we explain how to modify *Joint Story Model* to combine the story information and consider pairs jointly.

Single Story Model Given a story P and a set of participants in the story E, we create a separate input sequence based on every participant $e \in E$ following [21]:

$$C_{input} = [CLS]e[SEP]s_1[SEP]...s_n[SEP] \quad (1)$$

where $s_1, ...s_n$ are sentences in P. We then add timestep embeddings (0=padding or pseudo question, 1=past, 2=current, 3=future) [28] to mark the current reasoning step. The input embeddings and timestep embeddings are summed and encoded by the LM encoder. The [CLS] token representations from the input sequence of every timestep are extracted for output modeling, resulting in $C \in \mathrm{R}^{n \times d}$ where n is the number of steps in the story and d is the hidden dimension. We pass C to the attribute classifier, conflict detector, and story classifier for reasoning procedural components separately.

Attribute classifier is a typical two-layer feed-forward classification module that takes $C^i \in \mathrm{R}^d$ $(i \in n)$ in each sentence and predicts corresponding precondition and effect attributes. Given A attributes,

we include A precondition classifiers and A effect classifiers.

$$\theta_{attri} = W_a^T(tanh(W_d^T C^i)) \quad (2)$$

$$P_{attri} = agrmax(\theta_{attri}) \quad (3)$$

where $\theta_{attri} \in R^a$ and a is the number of possible labels of the attribute, e.g., *temperature* has three labels: *not related*, *cold*, and *hot*.

Conflict detector concatenates every pair of step representations $[C^i;C^j] \in \mathrm{R}^{2d}$ and processes them through a linear layer. Then, binary classification is applied to every sentence pair to predict whether it has conflicts.

$$C_{confl} = Stack(Concat(C^i, C^j)) \quad (4)$$

$$P_{confl} = Sigmoid(W_{confl} C_{confl}) \quad (5)$$

Story classifier computes the mean of the step representations $C^i \in \mathrm{R}^d$ to form a story representation $C_{sto} \in \mathrm{R}^d$ for the final label prediction. We project C_{sto} to a two-dimensional vector using a linear classifier to represent the plausible and implausible class logits. Thus, each story is classified separately.

$$C_{sto} = Mean(C) \quad (6)$$

$$P_{sto} = Softmax(W_{sto}^T C_{sto}) \quad (7)$$

Joint Story Model An obvious drawback of single-story modeling is that the relationship between the stories of the pair is not captured. However, since our model expects a unique input sequence for every participant in the story, it may be hard to construct parallel input sequences for stories that do not have identical participants. To remedy this, we first obtain the common participant set, $E_c = E \cap E'$, and then construct parallel input sequences for every $e \in E_c$ as equation [1]. Then for unaligned participants, we create a dummy participant e_0 to fill in the slot in the corresponding input sequence.

$$C_{dummy} = [CLS]e_0[SEP]s_1[SEP]...s_n[SEP] \quad (8)$$

Hence, both stories will have an equal number of participant-based input sequences (C for story P, C' for story P'). Both C and C' will still go through equations [2-5] to predict each story's participant attributes and conflict sentences. After obtaining C_{sto} and C'_{sto} from the pair of stories, we concatenate these two vectors and perform classification with a linear layer of size $\mathrm{R}^{d\times 1}$ to predict the plausible story. Hence both stories are jointly considered for prediction.

$$P_{sto} = W_{joint}^T C_{sto} \quad P'_{sto} = W_{joint}^T C'_{sto} \quad (9)$$

$$P_{joint} = Softmax(Concat(P_{sto}, P'_{sto})) \quad (10)$$

Both single and joint-story settings will give predictions on conflict sentences and story plausibility for each participant or participant pair. We take the mean of all participant predictions as the final prediction.

3.2 Training Regimes

As the model optimizes over multiple objectives, we experiment with story, participant, and sentence-centric training regimes.

Story Centric optimizes all three losses, i.e., story, conflict, and attribute loss, for each participant in the story.

$$L = L_{sto} + L_{confl} + \frac{1}{A}\sum_{i=0}^{A}(L^i_{prec} + L^i_{effe}) \quad (11)$$

Here we set the story label to implausible for all participants from the implausible story.

Participant Centric modifies the *story-centric* loss by setting the story label to implausible only for participants that appear in the conflicting sentences of the implausible story.

Sentence Centric omits the story loss, only optimizing the attribute and conflict losses.

$$L = L_{confl} + \frac{1}{A}\sum_{i=0}^{A}(L^i_{prec} + L^i_{effe}) \quad (12)$$

In the final story prediction of *sentence-centric loss*, we obtain the negative summation of all P_{confl} and detect which story is more plausible.

3.3 Augmentation Strategies

Procedural text understanding requires dense annotation of two procedural components in each step: (1) participants with relevant physical attributes, and (2) state labels for each physical attribute. Due to the laborious annotation requirement, most existing benchmarks are small in scale, which may hinder the learning of generalizable models. Instead of relying on manual annotation (e.g., by crowdsourcing), we propose a cheap and automatic data augmentation for training procedural reasoning models.

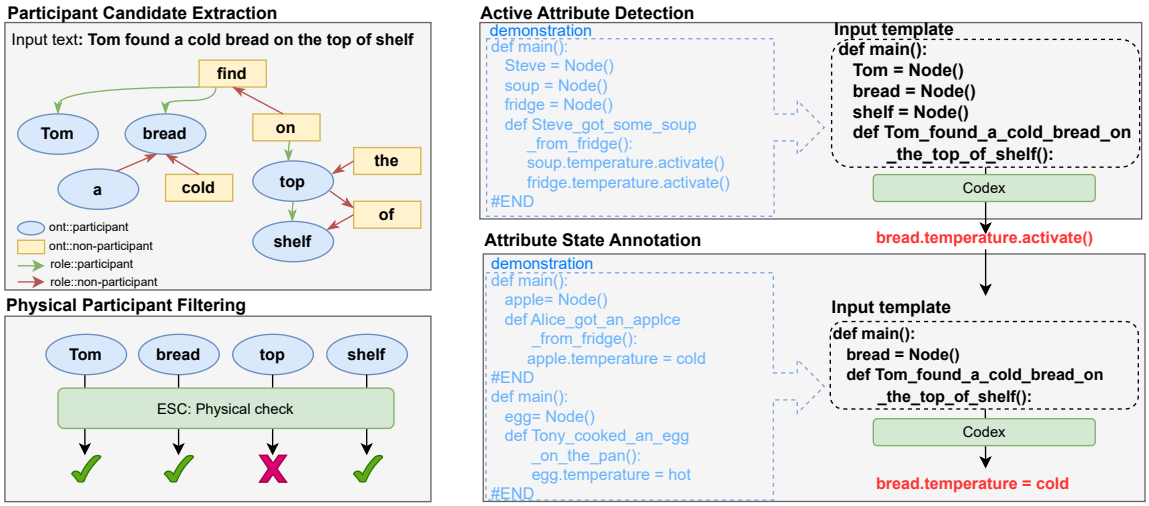
In-Domain Augmentation We define two in-domain methods: Participant Abstraction and Word Insertion. The *Participant Abstraction* augmentation is inspired by Höpner *et al.* [15]'s idea to replace specific concepts with more general ones to aid reinforcement learning. We replace all non-human participants with their direct hypernym in WordNet [23] to generate the new dataset. For example, "bread" $\rightarrow$ "baked goods", and "pan" $\rightarrow$ "cooking utensil". The intuition is that the hypernym shares similar physical properties with the original participant, enabling the reuse of the original attribute annotation. *Word Insertion* adds adjectives and adverbs to the existing sentences in the in-domain corpus. It selects suitable words based on a contextual word embedding of the original sentence, obtained with a pretrained LM, e.g., "Tom ate the cold soup" $\rightarrow$ "Tom ate the wonderful cold tomato soup". As word insertion merely enriches the participant information, we directly reuse the attribute annotation.

External Data Augmentation We experiment with two kinds of external data: *Natural Stories* and *Synthetic Stories*. As *Natural Stories*, we select ROCStories [25], a popular story cloze task dataset about everyday events. We use *Synthetic Stories* that are automatically generated from the Commonsense Knowledge Graph (CSKG) [17] based on psychological axioms [14]. The synthetic dataset, generated by [16] has over 100k plausible commonsense stories based on three story types with corresponding templates: unmet expectations, substitutions, and object modifications. Each plausible story is associated with an implausible story based on graph patterns. For our experiments, we use a stratified sample of 3K synthetic story pairs.

Unlike in-domain augmentation data, the external stories do not come with the dense attribute annotations assumed by LEAP. To bridge this gap, LEAP includes a novel labeler that can automatically extract participants and annotate their attributes without training, which we describe next.

4 Automatic Story Labeling

LEAP's labeler consists of *Participant Annotation* and *Attribute Annotation* (Figure 2). We annotate participants based on semantic rules



(a) Participant Annotation (b) Attribute Annotation

Figure 2: An overview of the LEAP's labeler. The labeler first annotates all participants in the story (a). The participants must belong to one of the pre-defined ontology classes, be involved in at least one semantic role relation, and be physical in nature. For each participant, we construct the corresponding Python function code and adapt few-shot prompting in Codex to label its attribute states automatically (b).

and verify each participant using WordNet. To annotate the attribute states of each participant, we devise a method for code-style few-shot LM prompting.

4.1 Participant Annotation

Participant Candidate Extraction Participants, including physical properties and phrases (e.g., birthday cake), lack a standardized detection method with high accuracy. We propose an innovative approach utilizing the TRIPS [1] parser's semantic information for participant identification. TRIPS generates logical forms for sentences, constructs a tiered ontology assigning words to specific classes, and identifies their semantic roles. We manually select eleven ontology classes and nine semantic roles (see appendix) and extract possible participants based on ontology class membership and semantic role involvement. Figure 2 (a) illustrates the exclusion of "cold" due to non-membership in ontology classes and "a" for lacking semantic role involvement. We combine participants sharing the same semantic roles in the sentence to form new participant phrases. This approach allows us to extract composite concepts like "dog cage", and even rare phrases like "guinea pig cage".

Physical Participant Filtering We then filter out non-physical participants because the annotated attributes only apply to physical ones. Notably, participant forms with ambiguous meanings can be physical in some contexts (e.g., light -> lamp) and non-physical in others (e.g., light -> sunshine). To enhance our method's generalizability, we use a word sense disambiguation model [2] to pick the appropriate word meaning in the story context. Finally, we check its physicality by traversing the WordNet hypernym tree of a synset iteratively until reaching the "physical entity" or "abstract entity" synset.

4.2 Attribute Annotation

For attribute annotation, we expect the model to provide participants' physical attribute states at every step of the story, which is essentially a structured prediction task. Inspired by recent findings that code-style prompts are more effective for structure-aware tasks [22], we expect that code format is more suitable than natural language in guiding the model to annotate stories. We propose to convert story inputs into Python-style prompts and leverage Codex [9] to generate the attribute annotation. As shown in Figure 2 (b), we convert the story prompt s into a Python function s^p, where each participant is defined as a Node class and each sentence in the story is converted to a function name. The attribute annotations a for a sentence are added as statements in the corresponding function. Then the evaluation story is converted to the same format to elicit outputs. Formally, for each evaluation story s_e to be annotated, the input to the prompting pipeline is:

$$a_e = Codex(s_1^p \oplus a_1 \oplus \cdots \oplus s_k^p \oplus a_k \oplus s_e^p) \quad (13)$$

where k is the number of demonstrations and the attribute annotation a_e is the output completion result of Codex.

A remaining challenge is that our stories have many participants, and we seek to extract a large set of attributes, resulting in large and sparse output space. In other words, only a few attributes for a subset of participants has non-trivial labels at any step of the story, while the rest are irrelevant, which makes the method by [22] insufficient for our case. To overcome the skewed label distribution, we propose a novel two-stage method that divides attribute annotation into *Active Attribute Detection* and *Attribute State Annotation*.

Active Attribute Detection As shown in Figure 2 (b), we first prompt Codex to detect the active participants among all possible participants at every step. Here, Codex sees $k = 4$ examples from in-domain data as prompt demonstrations and predicts the active participants for each attribute.

Attribute State Annotation For each active participant in the sentence, we perform another round of prompting to label its attribute state, as shown in the bottom right of Figure 2 (b). For attribute state annotation, We compute the word embedding of the participants and the sentence embedding of context for all possible demonstrations and pick the ones with the highest average cosine similarity. Then we apply Codex to generate the attribute state of the participant.

Table 1: Main evaluation results across five commonsense tasks with different data augmentation methods. We also included the zero-shot evaluation result of previous studies as well as the supervised setting. We reran TRIP and CGLI and obtained their zero-shot evaluation result as baselines. As TRIP, CGLI, and our model follow a participant-based input construction, we ran our labeler on five target tasks to extract participants. We report the average of three runs and their variation with different seeds. On the ROCStories dataset, we only use the CSKG portion of our augmentation data to respect the zero-shot setting - this result is marked with an asterisk (*) in the table.

Training	Model	Data size	In domain (TRIP)			Out of domain				
			Acc	Con	Ver	ROC	CODAH	PIQA	aNLI	RICA
Random	-	-	-	-	-	49.5	25.1	50.2	49.6	50.7
ZSQA	LM	-	-	-	-	70.2	49.5	67.6	65.5	50.3
	LM w/ CSKG	692K	-	-	-	89.7	68.5	**72.4**	70.5	51.7
TRIP	TRIP	0.8K	78.2	22.2	7.8	61.2	30.7	51.5	50.4	52.3
	CGLI	0.8K	94.1	77.3	28.0	76.9	43.3	54.4	60.0	49.6
	LEAP (no aug.)	0.8K	$97.3_{0.2}$	$\mathbf{78.4}_{1.0}$	$27.6_{1.2}$	$86.5_{0.3}$	$45.9_{1.7}$	$59.0_{0.7}$	$64.6_{1.5}$	$54.2_{0.4}$
	LEAP w/ labeler aug.	5.6K	$97.0_{0.7}$	$70.0_{2.8}$	$11.3_{3.3}$	$\mathbf{*90.6}_{0.4}$	$\mathbf{68.7}_{0.6}$	$68.6_{1.0}$	$\mathbf{71.8}_{0.9}$	$\mathbf{57.5}_{0.2}$
	LEAP w/ insertion aug.	1.6K	$\mathbf{97.8}_{0.3}$	$75.1_{1.3}$	$\mathbf{30.1}_{1.1}$	$85.1_{1.0}$	$43.9_{0.3}$	$57.7_{0.8}$	$63.5_{0.7}$	$54.6_{0.6}$
	LEAP w/ abstraction aug.	1.6K	$97.2_{0.7}$	$74.6_{3.1}$	$26.7_{3.4}$	$83.5_{0.9}$	$41.5_{0.9}$	$57.3_{0.4}$	$60.6_{0.8}$	$53.1_{0.9}$
Supervised	LM	-	-	-	-	$^{\dagger}97.9$	83.1	79.2	85.6	52.3

5 Experimental Setup

Datasets As a source task, we use the recently introduced TRIP [33] dataset, which contains 800 story pairs in total. Our target tasks are: 1) ROCStories [25], a story cloze task of selecting the plausible story ending out of two options. 2) PhysicalIQA (PIQA) [4], a two-choice question-answering dataset where the task is to pick the more plausible continuations out of two. 3) aNLI [3], where given the beginning and the ending of a story, the task is to choose the more plausible hypothesis out of two options. 4) CODAH [10], a multiple-choice sentence completion dataset where the task is to pick the most commonsensical choice. 5) RICA [40], a two-choice question-answering task of choosing a reasonable conclusion based on implicit commonsense relationships. We select these benchmarks because they represent a variety of tasks that can benefit from procedural reasoning. We provide further information about the datasets in the appendix.

Metrics We evaluate the procedural story understanding performance on the TRIP test dataset with three metrics: 1) accuracy of story classification, 2) consistency as a proportion of examples where both story and conflict sentences are correctly classified, and 3) verifiability as a proportion of examples with correct score on the first two metrics and with correct conflicting participant attributes in conflict sentences. For zero-shot evaluation, we report accuracy on the corresponding dev sets, as the test sets are not publicly available.

Baselines We compare against the following baselines. 1) As zero-shot QA (ZSQA) baselines, we consider the original *RoBERTa-Large* pre-trained model without any adaptation, and *RoBERTa-L (CSKG)* [20], which is adapted on 692K synthetic QA pairs generated from CSKG [17]. 2) We include baselines trained on the TRIP data with the same three losses introduced in section 3, namely, the original paper baseline [33] and *CGLI* [21]. 3) To contextualize the results, we also show the random baseline as a lower bound, and the supervised fine-tuned LM as an upper bound of zero-shot evaluation. For ROCstories, the supervised result is on the test dataset as training data is unlabeled and LM is trained on the dev set. We include implementation details in the appendix.

6 Results

Our experiments target six questions: *1) How does* LEAP *perform on in-domain tasks? 2) How does* LEAP *perform on out-of-domain tasks? 3) What is the optimal* LEAP *architecture for transferring knowledge to out-of-domain stories? 4) What is the effect of various augmentation strategies? 5) How does the* LEAP *labeling method compare to supervised labeling systems? 6) Does the* LEAP *attribute labeling and compositional design help explainability?*

Table 2: In-domain and out-of-domain zero-shot evaluation results of different LEAP models (modeling one[1] or jointly two stories[2]) and training losses (participant-, story-, and sentence-centric). The joint model makes the decision based on both story pairs and cannot be trained with participant-centric loss. Out-of-domain benchmarks: RS = ROCStories, CD = CODAH, QA = PIQA, aN = aNLI, RI = RICA.

LEAP	In domain (TRIP)	Out of domain				
Loss	Acc / Con / Ver	RS	CD	QA	aN	RI
$Part^1$	94.5 / 70.7 / 21.9	71.8	33.5	52.0	54.9	**54.2**
Sto^1	95.8 / 73.2 / 23.2	80.6	41.4	55.0	60.5	52.1
$Sent^1$	92.9 / 66.3 / 20.9	56.0	22.9	46.4	49.7	49.1
Sto^2	**97.3 / 78.4 / 27.6**	**86.5**	**45.9**	**59.0**	**64.6**	**54.2**
$Sent^2$	92.9 / 69.2 / 24.5	60.2	24.6	49.5	50.3	50.2

In-Domain Results On the in-domain task (Table 1), LEAP outperforms the procedural understanding baselines that are trained on the same data. Compared to the stronger baseline, CGLI, LEAP performs better in terms of accuracy and consistency, and on par in terms of verifiability. We also observe that story augmentation is detrimental to in-domain performance, which we attribute to the distribution shift of the additional data.

Out-of-Domain Results The results of the zero-shot transfer experiments (Table 1) show that LEAP outperforms the baselines, demonstrating the strong generalization achieved by model engineering and data augmentation. Despite using orders of magnitude fewer data for training, LEAP still outperforms *RoBERTa-L(CSKG)* on four out of five commonsense story tasks, showing the efficient data utilization of LEAP. We note that LEAP's performance is lower than [20] on PIQA and only slightly higher on CODAH. While this can be expected given the focus on QA of [20], we also hypothesize that the model performance is directly linked to the breadth of required knowledge. We compute the percentage of task participants that are unseen during training with the augmented data, observing that 66.7% of participants in PIQA are unseen, compared to 55.0% for CODAH and 43.7% for aNLI. This emphasizes the importance of suitable data augmentation and labeling, which is a key challenge addressed by our paper. We also note that although the story augmentation hurts the in-domain performance, it improves 3 to 23 absolute points across the datasets for out-of-domain evaluation. This is an

Table 3: Comparison of augmentation methods labeled with LEAP and CGLI. We report the average performance of three runs with variance.

LEAP		In domain (TRIP)			Out of domain				
Augmentation Data	**Labeler**	**Accuracy**	**Consistency**	**Verifiability**	**ROCStories**	**CODAH**	**PIQA**	**aNLI**	**RICA**
No augmentation	-	$97.3_{0.2}$	$\mathbf{78.4}_{1.0}$	$27.6_{1.2}$	$86.5_{0.3}$	$45.9_{1.7}$	$59.0_{0.7}$	$64.6_{1.5}$	$54.2_{0.4}$
CSKG	-	$95.7_{0.1}$	$45.1_{3.2}$	$17.4_{5.1}$	$89.6_{0.7}$	$57.9_{1.9}$	$66.0_{0.6}$	$67.4_{0.7}$	$55.5_{0.1}$
	CGLI	$95.9_{0.6}$	$69.1_{1.9}$	$24.5_{2.9}$	$90.0_{0.8}$	$61.6_{0.5}$	$66.1_{0.7}$	$68.1_{1.3}$	$53.2_{0.1}$
	LEAP	$96.2_{0.7}$	$68.9_{0.8}$	$16.0_{3.5}$	$\mathbf{90.6}_{0.4}$	$62.1_{1.1}$	$67.2_{0.8}$	$67.9_{1.0}$	$56.3_{0.1}$
ROCStories	-	$96.3_{0.4}$	$71.7_{6.7}$	$21.1_{4.1}$	-	$65.1_{2.5}$	$64.6_{0.6}$	$68.4_{0.2}$	$55.3_{0.7}$
	CGLI	$96.7_{0.3}$	$72.8_{1.8}$	$21.7_{1.4}$	-	$61.5_{0.5}$	$63.0_{0.3}$	$68.3_{0.1}$	$53.3_{0.1}$
	LEAP	$97.0_{0.4}$	$72.7_{0.6}$	$12.0_{1.6}$	-	$62.8_{1.0}$	$64.3_{1.0}$	$68.5_{0.3}$	$56.0_{0.1}$
CSKG+ROCStories	-	$95.3_{1.3}$	$33.6_{6.5}$	$8.9_{3.7}$	-	$\mathbf{68.7}_{0.9}$	$67.2_{0.7}$	$70.6_{1.1}$	$54.1_{0.8}$
	CGLI	$96.7_{1.4}$	$70.0_{3.0}$	$27.0_{2.5}$	-	$67.7_{0.7}$	$66.5_{0.3}$	$70.0_{0.8}$	$54.6_{0.3}$
	LEAP	$97.0_{0.7}$	$70.0_{2.8}$	$11.3_{3.3}$	-	$\mathbf{68.7}_{0.6}$	$\mathbf{68.6}_{1.0}$	$\mathbf{71.8}_{0.9}$	$\mathbf{57.5}_{0.2}$
Participant Abstraction	n/a	$\mathbf{97.8}_{0.3}$	$75.1_{1.3}$	$\mathbf{30.1}_{1.1}$	$85.1_{1.0}$	$43.9_{0.3}$	$57.7_{0.8}$	$63.5_{0.7}$	$54.6_{0.6}$
Word Insertion	n/a	$97.2_{0.7}$	$74.6_{3.1}$	$26.7_{3.4}$	$83.5_{0.9}$	$41.5_{0.9}$	$57.3_{0.4}$	$60.6_{0.8}$	$53.1_{0.9}$

Table 4: Labeling result on attribute states annotation.

Labeler	**Type**	**Prec.f1**	**Eff.f1**
TRIP	supervised	51.2	51.2
CGLI	supervised	72.1	75.6
LEAP	few-shot	61.2	70.3

interesting phenomenon that we explore further in later analysis.

Comparison of LEAP Architectures Table 2 shows the impact of story modeling and training regime choices within LEAP. The model architecture with story pairs and story-centric loss has the best performance on both in- and out-of-domain evaluations. *Joint* story modeling outperforms *single* story modeling in all metrics, indicating that the model benefits from considering the story pair input together and gaining insight from the direct comparison of stories. Among the different regimes, since models are able to handle the input globally, *participant-centric loss*, which provides fine-grained and local labels, can bring confusion and lead to a decline in performance. *Sentence-centric* loss has the worst performance for both story modeling options, and its zero-shot evaluation result is similar to or worse than random. *Sentence-centric* loss guides LEAP to find conflict sentence pairs in the story, which is reasonable for the in-domain task as each story pair must contain conflicting sentences but does not generalize well to out-of-domain tasks. Our analysis reveals that the incorrect answers on these out-of-domain tasks are often less commonsensical but not necessarily conflicting.

Effect of Augmentation As data augmentation helps with out-of-domain tasks and is harmful to in-domain tasks (cf. Table 1), we next investigate the impact of different augmentation strategies and labeling methods on these tasks. Here we use LEAP's labeler to extract participants from augmented stories and use either CGLI or LEAP to do annotation because CGLI does not perform participant annotation. The results in Table 3 show that story augmentation leads to lower results in the in-domain setting and improved out-of-domain accuracy regardless of the data partition or labeler, which is consistent with Table 1. Regarding data splits, CSKG and ROCStories perform comparably to each other as augmentation sources, whereas their combination reaches optimal performance, demonstrating the benefit of combining synthetic and natural data sources. Conversely, augmentation strategies that modify in-domain data increase the model's performance on the in-domain tasks but decrease its performance on the zero-shot evaluation tasks. These findings suggest that augmentation with additional data helps LEAP to generalize better to unseen stories, whereas modification strategies increase the overfitting to the in-domain data.

Comparison of Labeling Methods We also compare the labeling methods of CGLI and LEAP in Table 3. Here, we observe that using CGLI leads to better in-domain performance but LEAP outperforms CGLI on 12 out of 13 zero-shot evaluations. This indicates that CGLI fits the TRIP data distribution well, but may not generalize as well to new tasks. As CGLI uses the TRIP task to learn its attribute extractor, it is possible that it may also learn to fit the dataset artifacts or annotation errors. Meanwhile, LEAP leverages few-shot prompting to annotate new stories, which does not directly fit the task and is more generalizable to out-of-domain cases. Finally, augmentation without any fine-grained annotations (no labeler) leads to a drastic drop in in-domain performance and worse results on out-of-domain when using synthetic stories. This suggests that high-quality fine-grained labels are necessary for achieving robust procedural reasoning.

Intrinsic Evaluation of LEAP Labeler We measure the intrinsic performance of the two LEAP's labeler components on TRIP. For *Participant Annotation*, we compare against *spaCy*.[2] To extract participants with *spaCy*, we extract nouns from the story and combine them into possible noun phrases based on their location in the text. Compared to the gold participants in each story in TRIP, our labeler reaches 90.0% precision and 93.5% recall, while *spaCy* reaches 69.1% precision and 89.0% recall. This means that using *spaCy* directly leads to more noisy participant annotation. It also confirms that filtering participants based on semantic rules and physical properties is more effective. For *Attribute Annotation*, we compare our labeler with TRIP and CGLI on the in-domain task, and observe a gap in favor of the supervised methods (Table 4), as can be expected. Notably, prompting LLMs to solve the complex reasoning task directly has been popular in recent works [36], which found that Codex may fail to complete the reasoning path for complex stories requiring multiple-step reasoning or causality. In our case, we could have directly prompted Codex to solve the procedural reasoning task. Indeed, we see that Codex achieves decent attribute annotation performance on the TRIP dataset. Despite their promising results on these tasks, LLMs like Codex only have limited access through API calls, and they can be prohibitively expensive to run in many scenarios. Thus, we opt to use it for knowledge distillation, by annotating data that can transfer its knowledge to accessible models like RoBERTa.

Qualitative Analysis We conduct qualitative analysis to better understand the attribute labeling of CGLI and LEAP. Figure 3 shows the annotation of these two methods on one in-domain (from TRIP)

[2] https://spacy.io/

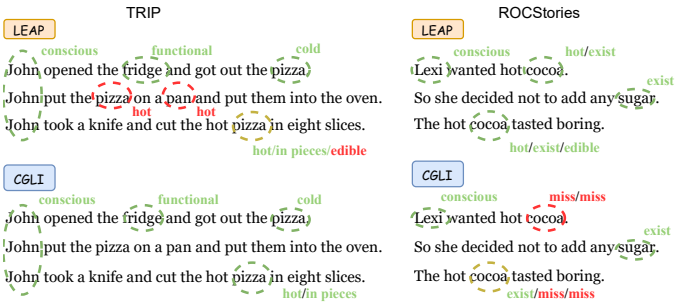


Figure 3: Case study of LEAP and CGLI attribute annotation on in- and out-domain tasks. All active participants are rounded in a dotted cycle with the actual attribute state. Green means correct labeling result, red means incorrect and yellow means the noise answer.

and one out-of-domain (ROCStories) story. For TRIP, we use the ground truth attribute annotation to evaluate the result, while for ROCStories, we manually evaluate the labeling result. For the in-domain story (Figure 3, left), all attribute states produced by CGLI are correct, whereas a small portion of LEAP's labels is incorrect. However, we observe that the labels produced by the LEAP 's labeler actually make sense even if they do not match the TRIP annotation, e.g., a *hot pizza* is definitely *edible*. This highlights annotation bias in TRIP that can be acquired and amplified by supervised models. On the zero-shot task, we observe that CGLI tends to label fewer participants compared to LEAP. In particular, in the last sentence, even though CGLI believes that *cocoa* exists, it still misses its *temperature* state. The lower generalizability of CGLI can be attributed to *cocoa* being an unseen participant for CGLI during training.

Besides reaching promising results on zero-shot evaluation, LEAP is also inherently interpretable through its tiered reasoning process. Consider the following story from ROCStories: *1) I decided to go on a bike ride with my brother. 2) We both headed out in the morning. 3) We were having a lot of fun. 4) Suddenly, he hit a rock and broke his wheel! 5) Watching my brother crash was fun.* For this story, our model infers the participant *wheel* is not functional after sentence 4, and that sentence 4 is in conflict with sentence 5. Based on this information, LEAP successfully classifies this story as implausible, demonstrating an interpretable and robust reasoning chain.

7 Discussion

Our experiments provide insights into the interplay of modeling architectures, training regimes, and augmentation strategies for both in- and out-of-domain reasoning over narratives. Our model, without any augmentation, reaches new state-of-the-art accuracy, consistency, and verifiability on an in-domain task. With augmentation, we are able to generalize much better on out-of-domain tasks, increasing the accuracy by 3 - 23 absolute points across datasets and performing better or competitive to prior baselines. Dense annotation of data either by our novel labeler or by recent work (CGLI) improves the effectiveness of the augmentation data. As our labeler is a few-shot system, it is able to generalize better to unseen stories. Finally, we observe that the joint modeling of stories with a compositional loss function brings the best performance. Our qualitative analysis shows that the better generalization of LEAP on out-of-domain tasks is accompanied by robust participant annotation and tiered reasoning.

With these insights in mind, we revisit three assumptions of our work and discuss future directions to improve on them:

Dynamic Selection of Augmentation Stories While our labeler can in theory be applied to any story, in this paper, we limit the augmentation stories to a single kind of synthetic stories (generated from a CSKG) and a single collection of realistic stories sourced from a popular corpus. Intuitively, the set of stories that can best benefit the model adaptation would depend on the downstream task, e.g., for a commonsense reasoning task, stories with procedural reasoning about household situations may benefit more than fables or fanfiction stories. While the Web provides an extensive collection of diverse stories, dynamic selection of stories for training augmentation has traditionally been extremely prohibitive due to the high costs of dense annotation. However, our few-shot labeler, enabled by SOTA techniques, opens the possibility for customizable collections of stories to be generated for tasks, or even subtasks. Prior work [37] has investigated sampling methods for a static collection of data; it is a key future work to investigate active learning strategies [39] for collecting and annotating augmentation data (semi-)automatically.

Comprehensive Labeling of Stories The participant states in this work are described with a relatively rich set of 20 physical attributes (e.g., temperature). As such, our model is largely geared toward modeling the physical world. Our current method can be expanded with complementary aspects of stories, such as the mental states of participants and causal links between events. The psychological axioms by Gordon and Hobbs [14] and the GLUCOSE dataset [26] with ROCStories event links provide a starting point for both directions, respectively. Moreover, all of our current attributes, except location, have binary (true/false, or high/low) values as states. Finer-grained annotation, e.g., with qualitative knowledge [13], can be considered in the future to improve the reasoning precision. The combination of manually created theories and resources with generic LM methods like our labeler provides an opportunity for a more comprehensive annotation of participant states posed as a generative task.

Comprehensive Understanding of Narratives Our evaluation in this paper follows the common practice of evaluating popular benchmarks with short stories, such as ROCStories, and TRIP. We believe that the development of methods that understand a wide range of narratives precludes the creation of a diverse set of story benchmarks, including fictional stories [27], and interactive story-driven games [34]. Curiously, as LM-based reasoning methods have been shown to rely on the surface similarity between the training and the test data (e.g., in terms of token length) [37], a general method would require the aforementioned dynamic selection of stories, but also novel methods that can abstract over surface properties better.

8 Conclusions

Our work devised LEAP: a framework for understanding stories through explainable procedural reasoning. LEAP integrates modeling architectures, training regimes, and augmentation strategies, selected with the aim to understand both in- and out-of-domain stories. LEAP alleviates the training data bottleneck with a novel labeling method that combines semantic parsing and structure-aware language models to annotate unseen stories in a robust manner. Our experiments showed that joint modeling of stories with a compositional loss function obtained new SOTA results on in-domain tasks. Augmentation with our labeler coupled with external natural or synthetic stories led to a significant increase in performance across out-of-domain tasks, showing strong generalization. Our preliminary experiments show that the labeler based on ChatGPT can perform worse than Codex, but the gap is not big. In the future, we will enhance LEAP with a dynamic selection of augmentation stories, increase the generality of the labeler by inferring mental states with fine-

grained values, and evaluate the explainability of LEAP with user studies on representative story tasks.

References

[1] James F. Allen and Choh Man Teng, 'Broad coverage, domain-generic deep semantic parsing', in *AAAI Spring Symposia*, (2017).

[2] Edoardo Barba, Tommaso Pasini, and Roberto Navigli, 'ESC: Redesigning WSD with extractive sense comprehension', in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics(ACL): Human Language Technologies*, pp. 4661–4672.

[3] Chandra Bhagavatula, Ronan Le Bras, Chaitanya Malaviya, Keisuke Sakaguchi, Ari Holtzman, Hannah Rashkin, Doug Downey, Wen-tau Yih, and Yejin Choi, 'Abductive commonsense reasoning', in *International Conference on Learning Representations(ICLR)*, (2019).

[4] Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi, 'PIQA: Reasoning about Physical Commonsense in Natural Language', in *Thirty-Fourth AAAI Conference on Artificial Intelligence*, pp. 7432–7439, (2020).

[5] Antoine Bosselut, Omer Levy, Ari Holtzman, Corin Ennis, Dieter Fox, and Yejin Choi, 'Simulating action dynamics with neural process networks', in *Proceedings of the 6th International Conference for Learning Representations (ICLR)*, (2018).

[6] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al., 'Language models are few-shot learners', *Advances in neural information processing systems(NeurIPS)*, **33**, 1877–1901, (2020).

[7] Eugene Charniak, *Toward a model of children's story comprehension.*, Ph.D. dissertation, Massachusetts Institute of Technology, 1972.

[8] Jiaao Chen, Jianshu Chen, and Zhou Yu, 'Incorporating structured commonsense knowledge in story completion', *Proceedings of AAAI*, **33**(01), 6244–6251, (Jul 2019).

[9] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al., 'Evaluating large language models trained on code', *arXiv preprint arXiv:2107.03374*, (2021).

[10] Michael Chen, Mike D'Arcy, Alisa Liu, Jared Fernandez, and Doug Downey, 'Codah: An adversarially-authored question answering dataset for common sense', in *Proceedings of the 3rd Workshop on Evaluating Vector Space Representations for NLP*, pp. 63–69, (2019).

[11] Yiming Cui, Wanxiang Che, Wei-Nan Zhang, Ting Liu, Shijin Wang, and Guoping Hu, 'Discriminative sentence modeling for story ending prediction', *Proceedings of AAAI*, **34**(05), 7602–7609, (Apr. 2020).

[12] Bhavana Dalvi, Lifu Huang, Niket Tandon, Wen-tau Yih, and Peter Clark, 'Tracking state changes in procedural text: a challenge dataset and models for process paragraph comprehension', *Proceedings of NAACL (Long Papers)*, (2018).

[13] Kenneth D Forbus. Qualitative reasoning., 1997.

[14] Andrew S Gordon and Jerry R Hobbs, *A formal theory of commonsense psychology: How people think people think*, Cambridge University Press, 2017.

[15] Niklas Höpner, Ilaria Tiddi, and Herke van Hoof, 'Leveraging class abstraction for commonsense reinforcement learning via residual policy gradient methods', in *IJCAI*, pp. 3050–3056. International Joint Conferences on Artificial Intelligence Organization, (2022).

[16] Filip Ilievski, Jay Pujara, and Hanzhi Zhang, 'Story generation with commonsense knowledge graphs and axioms', in *Workshop on Commonsense Reasoning and Knowledge Bases*, (2021).

[17] Filip Ilievski, Pedro Szekely, and Bin Zhang, 'Cskg: The commonsense knowledge graph', in *European Semantic Web Conference*, pp. 680–696. Springer, (2021).

[18] Tomáš Kočiský, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette, 'The NarrativeQA reading comprehension challenge', *TACL*, **6**, 317–328, (2018).

[19] Zhongyang Li, Xiao Ding, and Ting Liu, 'Story ending prediction by transferable bert', *Proceedings of IJCAI*, (Aug 2019).

[20] Kaixin Ma, Filip Ilievski, Jonathan Francis, Yonatan Bisk, Eric Nyberg, and Alessandro Oltramari, 'Knowledge-driven data construction for zero-shot evaluation in commonsense question answering', in *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 13507–13515, (2021).

[21] Kaixin Ma, Filip Ilievski, Jonathan Francis, Eric Nyberg, and Alessandro Oltramari, 'Coalescing global and local information for procedural

text understanding', in *Proceedings of the 29th International Conference on Computational Linguistics*, pp. 1534–1545, (2022).
[22] Aman Madaan, Shuyan Zhou, Uri Alon, Yiming Yang, and Graham Neubig, 'Language models of code are few-shot commonsense learners', in *EMNLP*, Abu Dhabi, UAE, (December 2022).
[23] George A. Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine J. Miller, 'Introduction to WordNet: An On-line Lexical Database*', *International Journal of Lexicography*, **3**(4), 235–244, (12 1990).
[24] Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer, 'Rethinking the role of demonstrations: What makes in-context learning work?', in *EMNLP*, (2022).
[25] Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen, 'A corpus and cloze evaluation for deeper understanding of commonsense stories', in *Proceedings of NAACL*, pp. 839–849, (June 2016).
[26] Nasrin Mostafazadeh, Aditya Kalyanpur, Lori Moon, David Buchanan, Lauren Berkowitz, Or Biran, and Jennifer Chu-Carroll, 'Glucose: Generalized and contextualized story explanations', in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 4569–4586, (2020).
[27] Thiloshon Nagarajah, Filip Ilievski, and Jay Pujara, 'Understanding narratives through dimensions of analogy', in *Workshop on Qualitative Reasoning (QR)*, (2022).
[28] Hossein Rajaby Faghihi and Parisa Kordjamshidi, 'Time-stamped language model: Teaching language models to understand the flow of events', in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics (ACL): Human Language Technologies*, pp. 4560–4570, Online, (June 2021). Association for Computational Linguistics.
[29] Laria Reynolds and Kyle McDonell, 'Prompt programming for large language models: Beyond the few-shot paradigm', in *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI EA '21, New York, NY, USA, (2021). Association for Computing Machinery.
[30] Yisi Sang, Xiangyang Mou, Jing Li, Jeffrey Stanton, and Mo Yu, 'A survey of machine narrative reading comprehension assessments', in *IJCAI 2022*, pp. 5580–5587. International Joint Conferences on Artificial Intelligence, (2022).
[31] Roger C Schank and Robert P Abelson, 'Scripts, plans, and knowledge', in *IJCAI*, volume 75, pp. 151–157, (1975).
[32] Evangelia Spiliopoulou, Artidoro Pagnoni, Yonatan Bisk, and Eduard Hovy, 'EvEntS ReaLM: Event reasoning of entity states via language models', in *Proceedings of EMNLP*, pp. 1982–1997, Abu Dhabi, United Arab Emirates, (December 2022). Association for Computational Linguistics.
[33] Shane Storks, Qiaozi Gao, Yichi Zhang, and Joyce Chai, 'Tiered reasoning for intuitive physics: Toward verifiable commonsense language understanding', in *Findings of EMNLP 2021*, (2021).
[34] Ruoyao Wang, Peter Alexander Jansen, Marc-Alexandre Côté, and Prithviraj Ammanabrolu, 'Scienceworld: Is your agent smarter than a 5th grader?', in *EMNLP*, (2022).
[35] Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le, 'Finetuned language models are zero-shot learners', in *ICLR*, (2021).
[36] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed H Chi, Quoc V Le, Denny Zhou, et al., 'Chain-of-thought prompting elicits reasoning in large language models', in *NeurIPS*.
[37] Jiarui Zhang, Filip Ilievski, Kaixin Ma, Jonathan Francis, and Alessandro Oltramari, 'A study of zero-shot adaptation with commonsense knowledge', *Automated Knowledge Base Construction(AKBC)*, (2022).
[38] Zhihan Zhang, Xiubo Geng, Tao Qin, Yunfang Wu, and Daxin Jiang, 'Knowledge-aware procedural text understanding with multi-stage training', in *WWW '21: The Web Conference 2021, Ljubljana, Slovenia, April 19–23, 2021*, (2021).
[39] Zhisong Zhang, Emma Strubell, and Eduard Hovy, 'A survey of active learning for natural language processing', in *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 6166–6190.
[40] Pei Zhou, Rahul Khanna, Seyeon Lee, Bill Yuchen Lin, Daniel Ho, Jay Pujara, and Xiang Ren, 'Rica: Evaluating robust inference capabilities based on commonsense axioms', in *Proceedings of EMNLP*, pp. 7560–7579, (2021).